



UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/074,030	02/14/2002	John R. Applin	10012812-1	5659

7590 11/01/2005
HEWLETT-PACKARD COMPANY
Intellectual Property Administration
P.O. Box 272400
Fort Collins, CO 80527-2400

EXAMINER

VU, TUAN A

ART UNIT PAPER NUMBER

2193

DATE MAILED: 11/01/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No. 10/074,030	Applicant(s) APPLIN ET AL	
	Examiner Tuan A. Vu	Art Unit 2193	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

- 1) ☒ Responsive to communication(s) filed on 15 September 2005.
- 2a) ☐ This action is **FINAL**. 2b) ☒ This action is non-final.
- 3) ☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

- 4) ☒ Claim(s) 1-21 is/are pending in the application.
- 4a) Of the above claim(s) _____ is/are withdrawn from consideration.
- 5) ☐ Claim(s) _____ is/are allowed.
- 6) ☒ Claim(s) 1-21 is/are rejected.
- 7) ☐ Claim(s) _____ is/are objected to.
- 8) ☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

- 9) ☐ The specification is objected to by the Examiner.
- 10) ☐ The drawing(s) filed on _____ is/are: a) ☐ accepted or b) ☐ objected to by the Examiner.
 Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).
 Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).
- 11) ☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

- 12) ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
- a) ☐ All b) ☐ Some * c) ☐ None of:
1. ☐ Certified copies of the priority documents have been received.
2. ☐ Certified copies of the priority documents have been received in Application No. _____.
3. ☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).
- * See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

- | | |
|---|---|
| 1) <input checked="" type="checkbox"/> Notice of References Cited (PTO-892) | 4) <input type="checkbox"/> Interview Summary (PTO-413) |
| 2) <input type="checkbox"/> Notice of Draftsperson's Patent Drawing Review (PTO-948) | Paper No(s)/Mail Date. _____ |
| 3) <input type="checkbox"/> Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) | 5) <input type="checkbox"/> Notice of Informal Patent Application (PTO-152) |
| Paper No(s)/Mail Date _____ | 6) <input type="checkbox"/> Other: _____ |

DETAILED ACTION

1. This action is responsive to the Applicant's response filed 9/15/2005.

As indicated in Applicant's response, claims 1, 9, 17 and 18 have been amended. Claims 1-21 are pending in the office action.

Claim Objections

2. Claims 1, 9, and 18 are objected to because of the following informalities: there appears to be a missing "the" between "version of each of" and "remaining object files" (li. 6, 7, 5, of respective claims). According to common grammar rules, any object noun followed by a 'of the' automatically implies and requires definiteness of the object (a relation linking any object to another via an 'of the' establishes definiteness of the former object bound by the 'of the') hence it cannot be without a preceding 'the' or a particular one thereof. Appropriate correction is required.

Claim Rejections - 35 USC § 101

3. 35 U.S.C. 101 reads as follows:

Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

The Federal Circuit has recently applied the practical application test in determining whether the claimed subject matter is statutory under 35 U.S.C. § 101. The practical application test requires that a "useful, concrete, and tangible result" be accomplished. An "abstract idea" when practically applied is eligible for a patent. As a consequence, an invention, which is eligible for patenting under 35 U.S.C. § 101, is in the "useful arts" when it is a machine, manufacture, process or composition of matter, which produces a concrete, tangible, and useful result. The test for practical application is thus to determine whether the claimed invention produces a "useful, concrete and tangible result".

4. Claim 17 is rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter.

Claim 17 is a non-method claim defining a software kit to facilitate application development and comprises modules, time stamp to correspond to elements of the kit, and a module to embed such time stamp in order to generate an error in response to a time stamp checking. As such all the elements claimed are construed as software elements in light of the specifications. The claim has thus no explicit teaching that would enable one skill in the art to be taught of any hardware utility or tangible support medium to embody and/or execute the software kit or the software modules as recited. Absent a hardware and tangible medium to support any such claimed elements, the claim amounts to a non-practical application as set forth in the Practical Application test above; hence is considered to be no more than an abstract idea for not yielding a concrete and tangible result; thus is rejected for leading to a non-statutory subject matter.

Claim Rejections - 35 USC § 112

5. The following is a quotation of the second paragraph of 35 U.S.C. 112:

The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

6. Claims 1-21 are rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

In claims 1, 9, 17, and 19, the limitation recited as ‘when the computer program is executed’ does not appear to associate a program being executed as commonly accepted and understood by one skill in the art in regard to compiler arts. As stated from the claim, this is a runtime environment wherein some object versions are compared; however, as the specifications point out (see *TimeStamp.h* - pg. 4-6; *pika.cc* – pg.7; *CheckVersion, compilation object* - pg. 6),

Art Unit: 2193

this runtime process is perceived as a compiling process since object files being fetched or parsed during the course of which are yet to be used to form/build (see pg. 10) a possible final executable code. This observation is based teachings --in the specification-- of the likes of .cc or .h files whose content/header is used by a preprocessor; and a compilation instance of a class like *CheckVersion* (see *compilation object* - pg. 6; pg. 4-8, *HB/how build, preprocessor textual* - pg. 10). As expected from common knowledge, when a program is executing would be construed as a definite executable code being a end result of some completed compilation process from interpreting the above claimed limitation. In light of the specifications, the instantiated *CheckVersion* class object derived from processing a .cc file in an environment wherein object files and headers are parsed or preprocessed only amounts at best to a linker/compiler process using some instantiated class methods. It is unclear from the claim as to whether 'program is executed' is really execution of program which has been a final product of compilation or just execution of a application/tool using instantiated object-oriented classes in a preprocessor environment to support a compilation tool and verifying of correctness of different object files being called upon to build a final executable. The limitation will be treated as though this execution (*runtime, computer program is executed*) is only a process executable to support comparing of versions of files fetched for use in broadly interpreted and various forms of compilation or software building processes, i.e. one stage of the compilation being performed. In short, the claimed invention as such is deficient in clarifying or defining whether runtime execution of a program is execution of a program being a result of compilation, or a program used during compilation or linking; and this is crucial because a runtime program executable is different from a executing compilation process.

Art Unit: 2193

Correction is required. The claims 2-8, 10-16, 19-21 are also rejected for not remedying to the indefiniteness of the base claims.

Claim Rejections - 35 USC § 103

7. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negated by the manner in which the invention was made.

8. Claims 1-16, 18-21 are rejected under 35 U.S.C. 103(a) as being unpatentable over Ching, USPN: 6,560,620 (hereinafter Ching) in view of Leblang et al., USPN: 5,649,200 (hereinafter Leblang).

As per claim 1, Ching discloses a method for verifying a version for each of a plurality of files in a computer program, the method comprising steps of:

identifying a version of a selected file of the plurality of files included in the computer program (col. 7, line 64 to col. 8, line 15);

comparing version of said selected file with version of each of the remaining files of the plurality of files (e.g. *with any selected second list* - col. 2, lines 40-42; col. 8, lines 45-49; col. 9, Fig. 7; line 61 to col. 10, line 24 – Note: Versioned sections of document being separately stored reads on plurality of versioned files – it is more meaningful to store a different content in separate file version than under a same version file -- being compared with the selected file, and category of same type being listed reads on comparing one selected file against plurality of same category of versioned files) when the computer program is executed (Note: file version being compared reads on an application being run); and

generating an alert in response to the version of said selected file being different than the one or more versions of the remaining files (e.g. *display on the user computer* - col. 2, lines 18-42).

But Ching does not explicitly disclose that the file being selected for comparing is a object file; however, Ching discloses a wide possibility to apply the comparing tool aiming at various the file content like programming languages (*software language ... arbitrary* - col. 15, lines 22-38). Leblang, in a tool to merge different version file being stored analogous to Ching, disclose object files (col. 1- 2; *msg.o, foo.o, bar.o* – Fig. 20, 24). In view of the same endeavor for detecting file differential by both Leblang and Ching, it would have been obvious for one of ordinary skill in the art at the time the invention was made to make the file type by Ching's tool so that it be also a object file as taught by Leblang because file as persisted can be reused in software application rebuilding (see Leblang: col. 1 line 66 to col. 2, line 42) and according to Ching's approach to alleviate application user effort (see col. 2), any file type that can be stored in computer (col. 7, lines 12-18) can be used in this comparing tool to alleviate extensive resource.

As per claim 2, Ching discloses a tool/kit to effect file comparison and Leblang discloses developers with merging, release building with auditing tools (see Leblang: Fig. 1, 6-7, 23); and in view of the rationale as set forth in claim 1 combining object file by Leblang, Ching discloses plurality of files includes a respective version related to software developer's kits (Fig. 7, 9 – Note: the software developer's kit for comparing object file has been rendered obvious with Leblang).

As per claim 3, in light of the rejection in claim 2, Ching identifying a version of an object file (Note: when the list of versioned file is displayed by hierarchy of version number, identifying version from a object file version is self-evident - Fig. 7, 9), storing the version of the selected file as a variable (Note: since this comparing tool is based on database, the persisting of file name therein for retrieval via code tool inherently teaches version defined in variable)

As per claim 4, in light of the rejection in claim 3, Ching the step of storing the list of object files presented for comparison as versions defined as variables used in the application code.

As per claim 5, Ching discloses generating an alert informing the user of a version mismatch (e.g. *display on the user computer* - col. 2, lines 18-42).

As per claim 6, see time stamp (col. 10, lines 18-21).

As per claim 7, Ching discloses displaying a respective textual message for the object file in response to the selected file being different than the version of one or more object files (see col. Fig. 5-8).

As per claim 8, Ching discloses storing the respective textual message of one of the plurality of files as an initial textual message in response determining the initial textual message equals a null value (Note: persisting file version with initial text belonging to very first, e.g. version 0 according to well-known practice - in memory/database to keep track of version is disclosed in Ching's teaching – see Fig. 2, Fig. 6, 7, 9); and displaying the initial textual message in response to the identified version being different than the initial version (refer to rejection of claim 7).

As per claim 9, Ching discloses a computer readable medium on which is embedded a program, the program performing a method for verifying a version for each of a plurality of files in a computer program, the method comprising steps of:

identifying (selected ... file ... plurality of files);
comparing (version ...selected ... file ... remaining ... plurality of files); and
generating (an alert ... version ... selected ... being different remaining ... plurality of files) when the computer program is executed (Note: file version being compared reads on an application being run); all of which steps having being addressed in claim 1.

But Ching does not explicitly teach that the selected file is an object file among a plurality of object files stored in a computer program. However, the object file being versioned has been addressed in the rationale as set forth in claim 1 using Leblang.

As per claims 10-16, these are computer medium claims corresponding to claims 2-8; hence are rejected with the rejections as set forth therein, respectively.

As per claim 18, Ching discloses an apparatus comprising means for:
identifying (selected ... file ... plurality of files);
comparing (version ...selected ... file ... remaining ... plurality of files); and
generating (an alert ... version ... selected ... being different remaining ... plurality of files) when the computer program is executed (Note: file version being compared reads on an application being run); all of which steps having being addressed in claim 1.

But Ching does not explicitly teach that the selected file is an object file among a plurality of object files stored in a computer program. However, the object file being versioned has been addressed in the rationale as set forth in claim 1 using Leblang.

Art Unit: 2193

As per claims 19-21, these are apparatus version claims corresponding to claims 2, 5, and 6; hence are rejected with the rejections as set forth therein, respectively.

9. Claims 17 is rejected under 35 U.S.C. 103(a) as being unpatentable over Bowman-Amuah, USPubN: 2001/0052108 (hereinafter Bowman) in view of Ching, USPN: 6,560,620.

As per claim 17, Bowman discloses a software development kit configured to facilitate a development of an application comprising a plurality of files, the software development kit comprises:

a plurality of modules (pg. 16, para 0471; para 0040; para 1172-1175 – Note: object oriented object or module and version tool reads on object file being stored);

a time stamp being utilized to identify plurality of file such as incident reports (*time stamps* -- para 2165-1272);

a version corresponding to a development software kit (*version control* – para 1703 – Note: build or release version in a versioning tool, wherein each release or build number reads on a software kit);

wherein the application developed with the software development kit is configured to generate response when detecting of a conflict caused by two of the files having different version (e.g. *version control* – para 1703; para 1172-1175, para 1180-1185 – Note: version control tool used by developers to synchronize time recorded versions of earlier in the version database – version of various builds or release or project/code modification levels - reads on generating a message informing the developer of such version difference in order to prevent conflicts – hence a error message -- in using non-synchronized version) when the computer program is executed (Note: file version being compared reads on an application being run).

But Bowman does not explicitly disclose time stamp to identify corresponding version of software kit for each plurality of object files; nor does Bowman disclose module configured to embed the time stamp within at least one of the plurality of files. It was well-known that version control for recording and tracking changes to files in a software development management environment is time related and even Bowman discloses this (see para 1175). Hence, having a time stamp to differentiate version of software file stored within version control repository is strongly implied or would have been obvious in view of such suggestion. Ching's tool for synchronizing software file discloses time stamp (col. 10, lines 13-23). In view of the endeavor to avoid conflicts as shown by Bowman, it would have been obvious for one of ordinary skill in the art at the time the invention was made to apply a time stamp to each of the versioned files in the Version Control database by Bowman because time-based parameter like time-stamp is known to be a differentiating attribute used in persisting data, files in many applications of the likes of Bowman's version control tool and Ching's repository of versioned files of a same category thus used in well-known version identifying endeavors.

Response to Arguments

10. Applicant's arguments filed 9/15/2005 have been fully considered but they are not persuasive. Following are the Examiner's response in regard thereto.

Claims rejection under 35 USC §103:

(A) Applicants have submitted that Ching and Leblang even if combined would yield a result that constitute a structure different from that of the claimed application and that the claimed invention is not about persisting files but to eliminate bugs when kits are used to compile programs (Appl. Rmrks, pg. 7, 2nd and 3rd para). There is no teaching in the claim that enforces

Art Unit: 2193

what appears to be a very strict limitation that the file version being compared is not for persisting but only for eliminating bugs by developer using software kits. In response to applicant's argument that the references fail to show certain features of applicant's invention, it is noted that the features upon which applicant relies (i.e., *eliminate bugs, software developer kits, not to have persistent versions, not to check ...that a file has been previously compiled* – see Appl. Rmrks pg. 7, middle) are not recited in the rejected claim(s). Although the claims are interpreted in light of the specification, limitations from the specification are not read into the claims. See *In re Van Geuns*, 988 F.2d 1181, 26 USPQ2d 1057 (Fed. Cir. 1993). The combination Ching and Leblang as set forth in the rejection has addressed each of the limitations claimed; and Applicants fail to point out precisely where the cited parts being put together would be inappropriate, notwithstanding the fact that a runtime program being executed as mentioned above is deemed unclear, i.e. Ching and Leblang teaching each an application which is also considered being executed in a runtime computer application environment.

(B) Applicants have submitted that the invention is to ‘assure that at computer program execution ... object files ... were all compiled ... kit or compiler’ (Appl. Rmrks pg. 7, bottom). A more thorough look at the recited elements of the claims shows that there is no explicit or remotely inferred and clear teaching about assuring that when a program is being executed all the object files are compiled using a same version of developer kit or compiler. In response to applicant's argument that the references fail to show certain features of applicant's invention, it is noted that the features upon which applicant relies (e.g. assure that object files were all compiled) are not recited in the rejected claim(s). Although the claims are interpreted in light of the specification, limitations from the specification are not read into the claims. See *In re Van*

Art Unit: 2193

Geuns, 988 F.2d 1181, 26 USPQ2d 1057 (Fed. Cir. 1993). Further, the above Applicant's arguments fail to comply with 37 CFR 1.111(b) because they amount to a general allegation that the claims define a patentable invention without specifically pointing out how the language of the claims patentably distinguishes them from the references. The Applicant is required to show inappropriate teaching from the reference cited parts in respect to each of the claimed limitation of the claim, as opposed to just contend with a general statement coming from what appears to have been gathered from an outside source of teaching rather than what is recited in the claim.

(C) Applicants have submitted that Bowman does not 'a time stamp used to identify ... development kit for each ... object files'; nor does Bowman teach 'means for determining whether ... uses object files compiled ... kits' (Appl. Rmrks pg. 8, middle). It is noted that Ching has been used to show why what is missing in Bowman would have been obvious based on specific suggestive teaching as set forth in the rejection. In response to applicant's arguments against the references individually, one cannot show nonobviousness by attacking references individually where the rejections are based on combinations of references. See *In re Keller*, 642 F.2d 413, 208 USPQ 871 (CCPA 1981); *In re Merck & Co.*, 800 F.2d 1091, 231 USPQ 375 (Fed. Cir. 1986).

(D) Applicants have submitted that neither Bowman or Ching as proposed would generate a similar result as the claimed invention and that neither teaches or suggests 'a software development kit configured to facilitate ... generate an error ... application is executed object files having ... different time stamps'; or that the combination as proposed would be capable of 'determining whether the program executable included object files ...compiled ...different software kits' (Appl. Rmrks pg. 8, bottom, pg. 9, top). The claimed software kit of claim 17

Art Unit: 2193

amounts to the following: a plurality of modules, a time stamp to identify a kit for each of the object files, a module to embed a timestamp in the object files, an executing program to generate an error when 2 object files have different time stamps. The issue of an executing program has been set forth earlier as to exhibit the deficiency as to how unclear this is perceived as put forth in the USC 112 rejection from above, i.e. the claim amounts to a process leading or building up to but not necessarily execution per se of an executable. The 103(a) rejection has shown that Bowman has a software kit or tool to identify when 2 versioned object files have different versions (e.g. build or release version in a versioning tool, wherein each release or build number reads on a software kit version) and Ching has been used to support the alert when there is discrepancy in time stamp in versioning tool similar to Bowman's software kit/tool. The rejection has also pointed out how Bowman's version control tool reads on showing version differences and possible conflicts in synchronized concurrent usage of versioned copies of files by plurality of developers; and has explained the self-evident teaching based thereon that upon detection of files version conflict a message to that effect is evident because a version tool is fundamentally a interactive tool for user to be notified of such mismatch (as via implied teachings or borrowing Ching's approach). And further, the rejection has also explained how time stamp is recognized as an attribute essential in establishing a unique characteristic used in differentiating time-based versions, with evidence in part from Bowman and also from Ching. Thus, the examiner recognizes that obviousness can only be established by combining or modifying the teachings of the prior art to produce the claimed invention particularly when there is some teaching, suggestion, or motivation to do so found either in the references themselves, or in the knowledge generally available to one of ordinary skill in the art. See *In re Fine*, 837

Art Unit: 2193

F.2d 1071, 5 USPQ2d 1596 (Fed. Cir. 1988) and *In re Jones*, 958 F.2d 347, 21 USPQ2d 1941 (Fed. Cir. 1992). The burden for Applicants is to show exactly how such combination would be prone to generating adverse effects or lead to negative teachings. Besides, asserting that the references fail to teach 'determine whether the program executable included object files that were compiled using different software developer kits' amounts to mere allegation of a patentable subject matter without considering the explicit elements being recited or the possible interpretations one skill in the art would derive therefrom. Applicant's arguments fail to comply with 37 CFR 1.111(b) because they amount to a general allegation that the claims define a patentable invention without specifically pointing out how the language of the claims patentably distinguishes them from the references.

For the above reasons, the claims stand rejected as set forth in the Office Action.

Conclusion

11. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Tuan A Vu whose telephone number is (272) 272-3735. The examiner can normally be reached on 8AM-4:30PM/Mon-Fri.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Kakali Chaki can be reached on (571)272-3719.

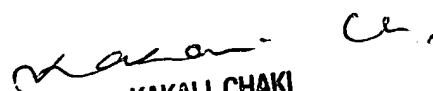
The fax phone number for the organization where this application or proceeding is assigned is (571) 273-3735 (for non-official correspondence – please consult Examiner before using) or 571-273-8300 (for official correspondence) or redirected to customer service at 571-272-3609.

Art Unit: 2193

Any inquiry of a general nature or relating to the status of this application should be directed to the TC 2100 Group receptionist: 571-272-2100.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see <http://pair-direct.uspto.gov>. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

VAT
October 24, 2005


KAKALI CHAKI
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100